KANDIDAATINTYÖ
Hiiren käyttäminen kamerana

Viljami Käsmä
Ohjaaja: Timo Rahkonen

ELEKTRONIIKAN JA TIETOLIIKENNETEKNIIKAN TUTKINTO-OHJELMA

2020

Käsmä V.A. (2020) Hiiren käyttäminen kamerana.
Oulun yliopisto. Elektroniikan ja tietoliikennetekniikan tutkinto-ohjelma. Kandidaatintyö, 22 s.

TIIVISTELMÄ

Tässä kandidaatintyössä tutustuttiin optisen Logitech M-BJ67B -hiiren ja erityisesti sen Agilent ADNS-2051 -sensorin toimintaan. Käytännön työnä hiirestä modifioitiin kamera, eli luettiin ja visualisoitiin hiiren sensorin raaka sävykuva. Lopuksi pohdittiin, voisiko ADNS-2051-sensoria soveltaa muilla tavoin. Motivaationa työn tekemiselle oli selvittää optisen hiiren ja sen sensorin toimintaperiaate ja samalla arvioida optisen sensorin soveltuvuutta muihin käyttötarkoituksiin.
Teoriaosuudessa tutkittiin M-BJ67B-hiiren ja ADNS-2051-sensorin toimintaa pohjautuen pääosin sensorin datalehteen. Lisäksi käytiin läpi sävykuvan lukeminen sensorilta ja tämän tekevän Arduino-ohjelman toiminta. Käytännön työ tehtiin purkamalla hiiri ja yhdistämällä sen ADNS-2051-sensori Arduino UNO -mikrokontrolleriin, joka oli ohjelmoitu lukemaan sävykuva sensorilta. Viimeisenä vaiheena sävykuvamatriisi visualisoitiin tietokoneella.
Työn lopputuloksena sävykuva saatiin onnistuneesti näkyviin ja sensorin arvioitiin soveltuvan ainakin vedenlaadun tarkkailuun ja liiketunnistimeksi. Loppupäätelmänä kuitenkin todettiin olemassa olevan tutkimuksen pohjalta, että muita mahdollisia käyttötarkoituksia on lukuisia.

Avainsanat: kamera, Arduino, optinen hiiri, sensori, sarjaportti, sävykuva, skanneri, liikesensori

Käsmä V.A. (2020) Using a mouse as a camera.
University of Oulu. Degree Programme in Electronics and Communications Engineering, 22 p.

ABSTRACT

The objective of this thesis was to explain the operating principle of an old optical mouse, Logitech M-BJ67B, and its sensor, Agilent ADNS-2051. In the practical part of the thesis the mouse was modified to be used as a camera i.e., the raw image data seen by the sensor was extracted and visualized. Other possible uses for the mouse sensor were also discussed. Motivation for the thesis was to find out the operating principle of an everyday item and to assess if the mouse sensor could be adapted to different use cases.

The theoretical part of the thesis reviewed the operation of the M-BJ67B mouse, and its ADNS-2051 sensor based primarily on the data sheet of the sensor. In addition, the process of reading the raw image data from the sensor and the operation of the Arduino program used for this were reviewed. The practical work was done by disassembling the mouse and connecting its ADNS-2051 sensor to the Arduino UNO microcontroller which had been programmed to read the pixel dump image from the sensor. The last step was visualizing the pixel dump matrix on a computer.

The result of the work was successful displaying of the pixel dump image and the estimation that the mouse sensor would be suitable for at least water quality monitoring and motion detection. However, it was further concluded by reviewing existing research on the subject that many more applications are possible.

Keywords: camera, Arduino, optical mouse, sensor, serial port, scanner, motion sensor

ALKULAUSE

Motivaatio aiheen valitsemiseen lähti mielenkiinnosta tutustua jokapäiväisen laitteen toimintaan. Työn tekeminen opetti minulle paljon ja toivon, että myös työn lukija oppii siitä jotain hyödyllistä tai ainakin mielenkiintoista.

Oulussa, syksyllä 2020

Viljami Käsmä

SISÄLLYS

LYHENTEET

AGS Automatic Gain Control.

CPI counts per inch.

DPI dots per inch.

DSP Digital Signal Processor.

FPS frames per second.

GND System ground.

IAS Image Acquisition System.

IPS inches per second.

MSB Most Significant Bit.

PD Power Down.

SCLK Serial port clock (input).

SDIO Serial data (input and output).

SQUAL Surface Quality.

USB Universal Serial Bus.

$\mathbf{V_{DD}}$ 5.0 volt power supply.

1. JOHDANTO

Mekaaninen tietokonehiiri kehitettiin 1960-luvulla. Ensimmäisten mallien pohjassa oli kaksi potentiometreihin kiinnitettyä kohtisuorassa toisiaan vastaan olevaa pyörää. Pyörät korvattiin myöhemmissä malleissa pallolla ja potentiometrit kulma-antureilla, mikä mahdollisti hiiren vapaamman liikuttamisen. Ensimmäiset optiset mallit kehitettiin 1980-luvulla, mutta ne vaativat vielä erityisen alustan toimiakseen. Ensimmäiset varsinaiset optiset hiiret eli pientä kameraa sensorina käyttävät mallit tulivat markkinoille 90-luvun lopussa. Nykyisten hiirten toimintaperiaate ei ole muuttunut sitten 90-luvun, mutta sen sijaan sensorien suorituskyky on parantunut merkittävästi ja ominaisuuksien määrä on kasvanut.[1–3]

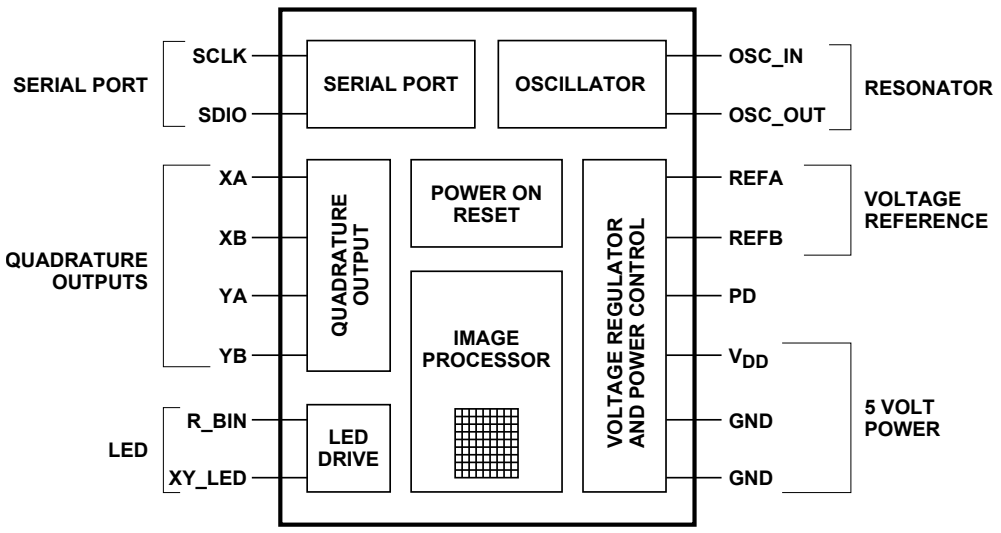
Tietokoneiden yleistyttyä hiiren merkitys syöttölaitteena on muuttunut kriittiseksi. Nykyisin optinen hiiri on monelle arkipäiväinen laite, ja sen toimintaa pidetään itsestäänselvyytenä. Tämän työn tavoitteena onkin dokumentoida vanhan optisen hiiren toiminta ja tutkia, miten hiiren kamerasensorin kuvadata saadaan näkyviin. Lopuksi pohditaan vielä sensorille muita mahdollisia käyttötarkoituksia.

2. TYÖN TEORIA

2.1. Optisen hiiren toimintaperiaate ja rakenne

Tässä työssä tutustutaan Logitech M-BJ67B -hiiren toimintaan. Hiiri käyttää vuonna 2003 julkaistua optista Agilent ADNS-2051-sensoria. Sensori ei toimi itsenäisesti vaan vaatii sitä hallitsevan mikrokontrollerin. Hiiressä käytettävä mikrokontrolleri on Cypress CP5928AM, joka kommunikoi tietokoneen kanssa USB-liitännän välityksellä. CP5928M-mikrokontrolleri on patentoitu, joten siitä ei ole saatavissa sen toimintaa kuvaavaa datalehteä.

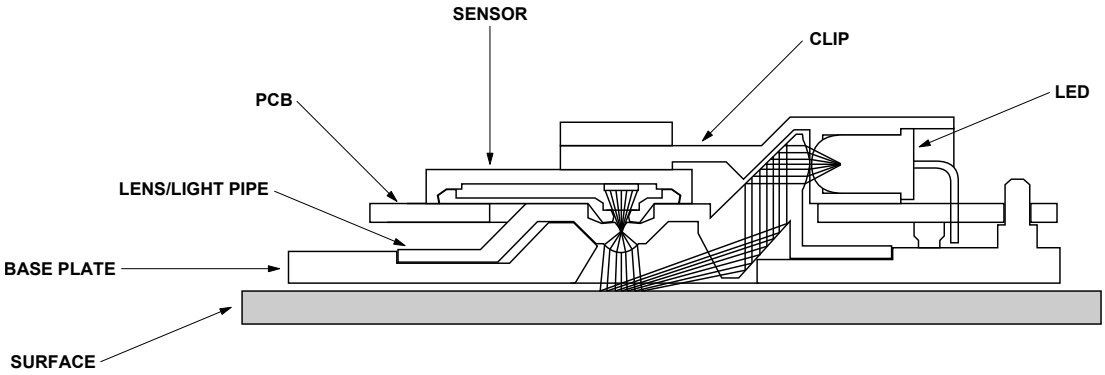
A2051-sensori on pieni 16x16-resoluution kamera, joka ottaa mustavalkokuvia hiiren alla olevasta pinnasta 1500 kertaa sekunnissa. Edellinen kuva tallennetaan sensorin sisäänrakennettuihin rekistereihin, jolloin sensorin Digital Signal Processor (DSP) -lohko voi verrata nykyistä ja edellistä kuvaa toisiinsa ja laskea niiden päällekkäisyydestä, minne hiirtä on liikutettu ja kuinka paljon. Sensori ilmoittaa muutoksen suhteellisina siirtyminä Δx ja Δy, jotka ovat luettavissa sensorista kahdella tavalla: suorakulmaisena kaksikanavaisena signaalina pinneistä XA, XB, YB ja YA tai sarjaportin kautta eli Serial data (input and output) (SDIO) -pinnistä. CP5928AM-mikrokontrolleri käyttää jälkimmäistä tapaa ja lähettää siirtymät tietokoneelle USB:n välityksellä, jossa ne näkyvät kursorin liikkeenä näytöllä.[2]



Kuva 1. Agilent ADNS-2051 lohkokaavio.

Hiiren alla oleva pinta on valaistava sekä tarkennettava luotettavan seurannan mahdollistamiseksi. Pinta valaistaan hiiren sisällä olevalla ledillä, joka voi olla

erillinen hiiren piirilevyyn liitetty komponentti tai itse sensoriin integroitu. Tämän työn tapauksessa ledi on erillinen komponentti, kuten kuvasta 2 nähdään. Ledin aallonpituus voi olla näkyvän tai infrapunavalon alueella riippuen, mille aallonpituudelle hiiren sensori on herkin. ADNS-2051-sensorin suhteellinen vaste on korkeimmillaan punaisen valon alueella, jonka vuoksi hiiressä käytettävä ledi on punainen. HDNS-2100-linssi käyttää kokonaisheijastuksia hyödyntävää "valoputkea" ledin valon ohjaamiseen sensorin alapuolella olevalle pinnalle valaisun kannalta optimaalisessa kulmassa. Valo heijastuu pinnasta takaisin linssiin, joka tarkentaa sen matkalla sensorille. HDNS-2100-linssi toimii parhaiten, kun matka linssin referenssitasosta eli jalustasta seurantapinnalle on 2.1-2.7 mm. Kuvassa 2 nähdään ADNS-2051 -sensorin suositeltu kokoonpano ja valon reitti ledistä sensorille.[1–3]



Kuva 2. Agilent ADNS-2051 sensorin suositeltu kokoonpano.

Laserhiiri on optisen hiiren erikoistapaus, jossa käytetään normaalin ledin sijasta laserdiodia. Koherentti laservalo tuottaa korkean kontrastin kuvan, joka mahdollistaa pinnan yksityiskohtien erottamisen myös heijastavilta ja läpinäkyviltä pinnoilta kuten lasilta tai metallilta. Tämän seurauksena laserhiirtä voi käyttää pinnoilla, joilla optinen hiiri ei muuten toimisi.[1]

2.2. Suorituskykytermistöä

Sensorin kuvanottonopeus ilmoitetaan frames per second (FPS) -lukuna, joka vaihtelee tyypillisesti välillä 500 - 7000 FPS. FPS-luku vaikuttaa suoraan sensorin suurimpaan mahdolliseen seurantanopeuteen, joka mitataan yksikössä inches per second (IPS).[1] Nykypäivän suorituskykyisimmät sensorit kykenevät ottamaan jopa 12000 kuvaa sekunnissa ja seuraamaan hiiren liikettä, jonka nopeus on 250 ips (6,35m/s). Vertailun vuoksi ADNS-2051-sensori ei pysty seuraamaan liikettä, joka on nopeampaa kuin 14 ips (35,56 cm/s).[3]

2.3. ADNS-2051-sensorin sarjaporttitoiminta

ADNS-2051-sensorin ja Arduinon väliseen kommunikaatioon voidaan käyttää sarjaporttikommunikaatiota. Sarjaportti vaatii kaksi johtoa: datalinjan (SDIO) itse tiedonsiirtoon ja kellon (Serial port clock (input) (SCLK)) synkronoimaan tiedonsiirtoa. Data voi kulkea kumpaan suuntaan tahansa, mutta ei yhtä aikaa. Isäntänä toimiva mikrokontrolleri aloittaa aina kommunikaation.

Kirjoitusoperaatiossa mikrokontrolleri lähettää kaksi tavua sensorille. Ensimmäisen tavun eniten merkitsevä bitti eli Most Significant Bit (MSB) on "1", joka merkitsee datan suuntaa. Loput seitsemän bittiä ovat rekisteriosoite, johon toisessa tavussa oleva varsinainen data kirjoitetaan. Mikrokontrolleri muuttaa SDIO:a SCLK:n laskevalla reunalla ja ADNS-2051 lukee sen nousevalla reunalla. Kirjoitusoperaation jälkeen täytyy olla vähintään 100 mikrosekunnin viive (t_{SWW}, t_{SWR} $\geq$ 100 μs) ennen seuraavaa operaatiota, jotta kirjoitusoperaation oikea toiminta varmistuu.

Lukuoperaatio koostuu myös kahdesta tavusta. Ensimmäinen tavu sisältää kirjoitusoperaation tapaan rekisteriosoitteen, mutta nyt MSB on "0"eli datan suunta on sensorilta mikrokontrollerille. Seuraavaksi ADNS-2051 vaatii ainakin 100 mikrosekunnin viiveen ($t_{HOLD} \geq 100$ μs) datan valmistelemiseen ennen seuraavaa kellojaksoa. Tämän jälkeen ADNS-2051 alkaa vaihtaa SDIO:n tilaa SCLK:n laskevalla reunalla ja mikrokontrolleri lukee sen SCLK:n nousevalla reunalla. Lukuoperaation jälkeen seuraavaa kellojaksoa täytyy viivyttää vähintään 120 nanosekuntia (t_{SRW}, $t_{SRR} \geq 100$ ns) ennen seuraavaa luku- tai kirjoitusoperaatiota.[2]

ADNS-2051-sensorin sarjaporttikommunikaatioon liittyy myös kolmas pinni SDIO:n ja SCLK:n lisäksi. Power Down (PD)-pinniä käytetään sarjaporttikommunikaation uudelleensynkronoimiseen ja sensorin asettamiseen virransäästötilaan. Mikrokontrolleri voi nostaa PD-pinnin 100 μs:n ajaksi ylös, jolloin ADNS-2051-sensori nollaa sarjaportin, mutta ei rekistereitä.[2]

2.4. ADNS-2051-sensorin rekisterit

Sarjaportin avulla päästään käsiksi ADNS-2051-sensorin rekistereihin, joiden kautta voidaan ohjelmoida sensoria sekä lukea asetuksia ja muuta dataa. Kuvassa 3 nähdään kaikki 18 rekisteriä ja niiden toimintaa kuvaavat nimet. Rekisterien osoitteet on ilmoitettu heksadesimaalilukuina välillä 0x00-0x11. Kaikkien rekisterien arvot on mahdollista lukea, mutta vain seuraaviin rekistereihin voidaan myös kirjoittaa: Configuration_bits, Frame_Period_Lower ja Frame_Period_Upper.[2]

Address	Register	Address	Register	Address	Register
0x00	Product_ID	0x06	Average_Pixel	0x0c	Data_Out_Lower
0x01	Revision_ID	0x07	Maximum_Pixel	0x0d	Data_Out_Upper
0x02	Motion	0x08	Reserved	0x0e	Shutter_Lower
0x03	Delta_X	0x09	Reserved	0x0f	Shutter_Upper
0x04	Delta_Y	0x0a	Configuration_bits	0x10	Frame_Period_Lower
0x05	SQUAL	0x0b	Reserved	0x11	Frame_Period_Upper

Kuva 3. Agilent ADNS-2051-sensorin rekisterit.[2]

2.5. ADNS-2051-sensorin sävykuvan lukeminen

ADNS-2051-sensorissa on sisäänrakennettu toiminto sävykuvan lukemiseen sarjaportin kautta. Configuration_bits-rekisterin kolmannen bitin eli PixDump-bitin muuttaminen ykköseksi kytkee päälle sensorin Pixel Dump -toiminnon. Toiminnon ollessa päällä sensori “dumppaa” sävykuvan Data_Out_Lower- ja

Data_Out_Upper-rekisterien kautta. Data_Out_Upper-rekisteri sisältää pikselin osoitteen ja Data_Out_Lower-rekisterin kuusi vähiten merkitsevää bittiä sisältävät pikselin arvon. Data_Out_Lower-rekisterin MSB on datan statusbitti, jonka arvo kertoo, onko rekisterin data validia. Data on validia, mikäli bitti on 0.

Sävykuvan 16x16 eli 256 pikseliä dumpataan rekistereihin peräkkäin yksi kerrallaan. Dumppaus aloitetaan sävykuvamatriisin osoitteesta 0x00, jonka jälkeen jokaisella seuraavalla pikselillä osoitetta kasvatetaan yhdellä, kunnes päästään viimeiseen osoitteeseen 0xFF. Sävyarvolla on käytössä 6 bittiä, joten pikselin valoisuuden maksimiarvo on $2^6 - 1 = 63$. Tosin sensorin sisäänrakennettu Automatic Gain Control (AGS) -piiri säätää suljinarvoa siten, että kuvan kirkkain arvo on noin 55.[2]

2.6. Arduino-ohjelman toiminta

Sävykuvan lukevan mousecam-ohjelman[4] (ks. Liite 1) toiminta perustuu pitkälti edellisissä osioissa läpikäytyihin operaatioihin, jotka on toteutettu omina funktinoinaan Arduino-ohjelmakoodiin. Luku- ja kirjoitusoperaatioilla sekä PixDump- ja Reset-toiminnoilla on omat funktionsa.

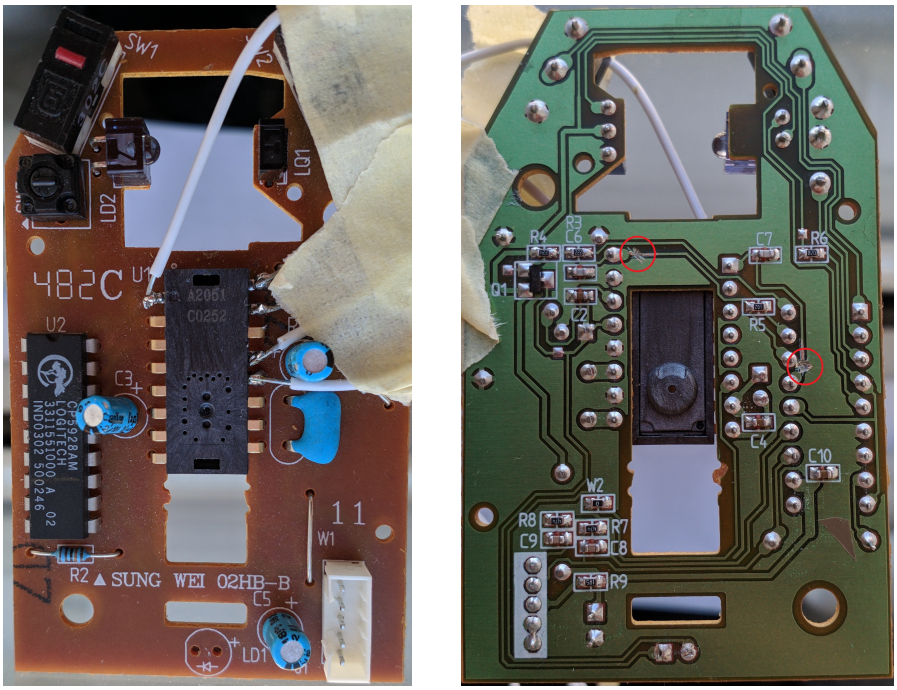
Arduino-ohjelman "sydän" on jatkuvasti suoritettava loop-funktio, joka kutsuu kaikkia muita funktioita ja lukee sävykuvaa. Aluksi loop-funktio tarkistaa, onko sarjaporttiyhteyttä saatavilla, jolloin luetaan dumpWidth-parametri sävykuvaa visualisoivalta javaohjelmalta. Parametri on oletuksena täydet 256 pikseliä. Seuraavaksi luetaan REG_MOTION-rekisteri, mikä jäädyttää Δx- ja Δy-arvot sisältävät rekisterit, kunnes ne tai REG_MOTION-rekisteri luetaan. Luetaan liikerekisterit ja tulostetaan arvot sarjaportilla. Viimeiseksi kutsutaan dumpFrame-funktiota, mikäli dumpWidth-parametria ei ole säädetty nollaksi.[4]

3. TYÖN TOTEUTUS JA TULOKSET

Tässä luvussa käydään läpi teknisen työn toteutus ja tulokset.

3.1. Logitech M-BJ67 -hiiren modaus

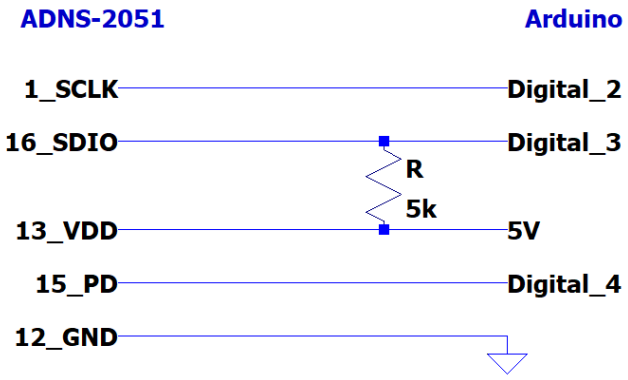
Työssä käytettiin Arduino UNO -mikrokontrolleria ja Logitech M-BJ67B -hiirtä, jossa on Avago ADNS-2051 -sensori. Aloitettiin avaamalla hiiri, poistamalla piirilevy hiiren sisältä ja irrottamalla piirilevystä kaikki tarpeeton kuten USB-johto ja rulla. Seuraavaksi katkaistiin sensorin ja mikrokontrollerin väliset SDIO- ja SCLK-johtimet, jotta mikrokontrolleri ei häiritsisi Arduinon toimintaa.



(a) Piirilevyn yläpuoli. (b) Piirilevyn alapuoli.

Kuva 4. Piirilevy ja kolvaukset. Kuvaan b on merkitty punaisilla ympyröillä kohdat, joista johtimia on katkaistu.

Nyt voitiin juottaa kiinni viisi johdinta sensorin jalkoihin helppoa Arduinoon yhdistämistä varten. Johtimet juotettiin kiinni seuraaviin pinneihin: 1 SCLK, 12 GND, 13 V_{DD}, 15 PD ja 16 SDIO. Johtimet yhdistettiin sitten kuvan 5 mukaan Arduinoon.[4] SDIO yhdistettiin myös käyttöjännitteeseen 5 $k\Omega$:n vastuksella. Vastus on passiivinen ylösvetovastus, jolloin vastus vetää jännitteen ylös ja Arduino tai sensori voi vetää jännitteen alas.



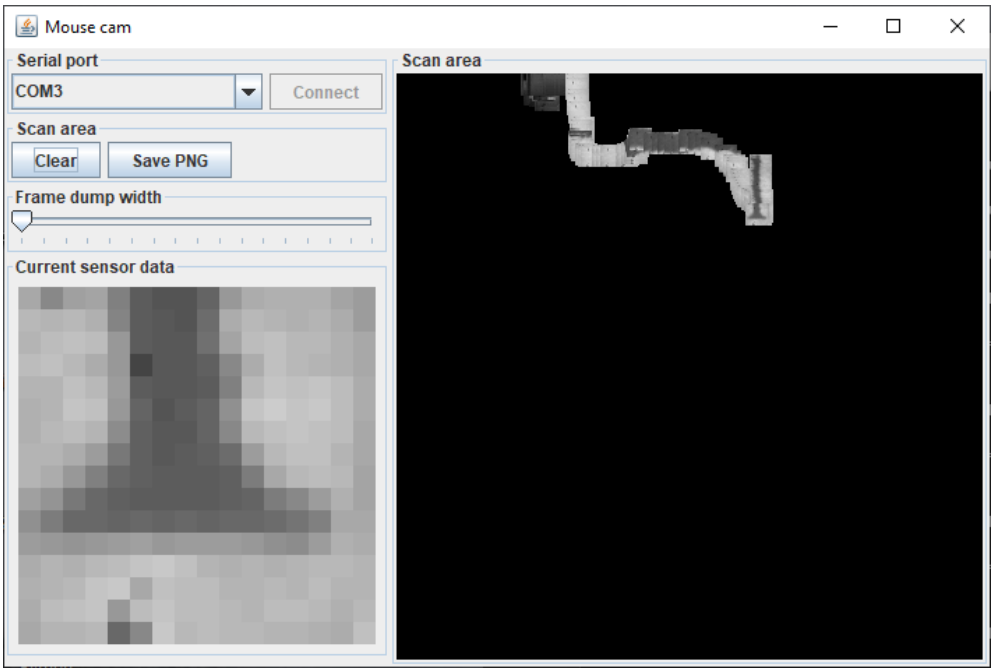
Kuva 5. Sensorin ja Arduinon välinen kytkentä havainnollistettuna LTSpicellä.

3.2. Arduinon ohjelmointi ja sävykuvan visualisointi

Tarpeellisten rautamodifikaatioiden jälkeen Arduino yhdistettiin tietokoneeseen ja siihen lähetettiin sävykuvaa lukeva ohjelma (ks. liite 1) Arduino IDE -ohjelmistolla. Seuraavaksi asennettiin tietokoneelle (Windows) RXTX-kirjasto[5], joka lisää Javaan sarjaporttituen. Kommunikaatio tietokoneen ja Arduinon välillä tapahtuu sarjaportilla. Nyt voitiin avata terminaali, navigoida oikeaan kansioon ja käynnistää graafinen mousecam-javaohjelma alla olevalla komennolla. Ohjelman suorittamista varten tietokoneessa täytyy olla asennettuna 32-bittinen Java.

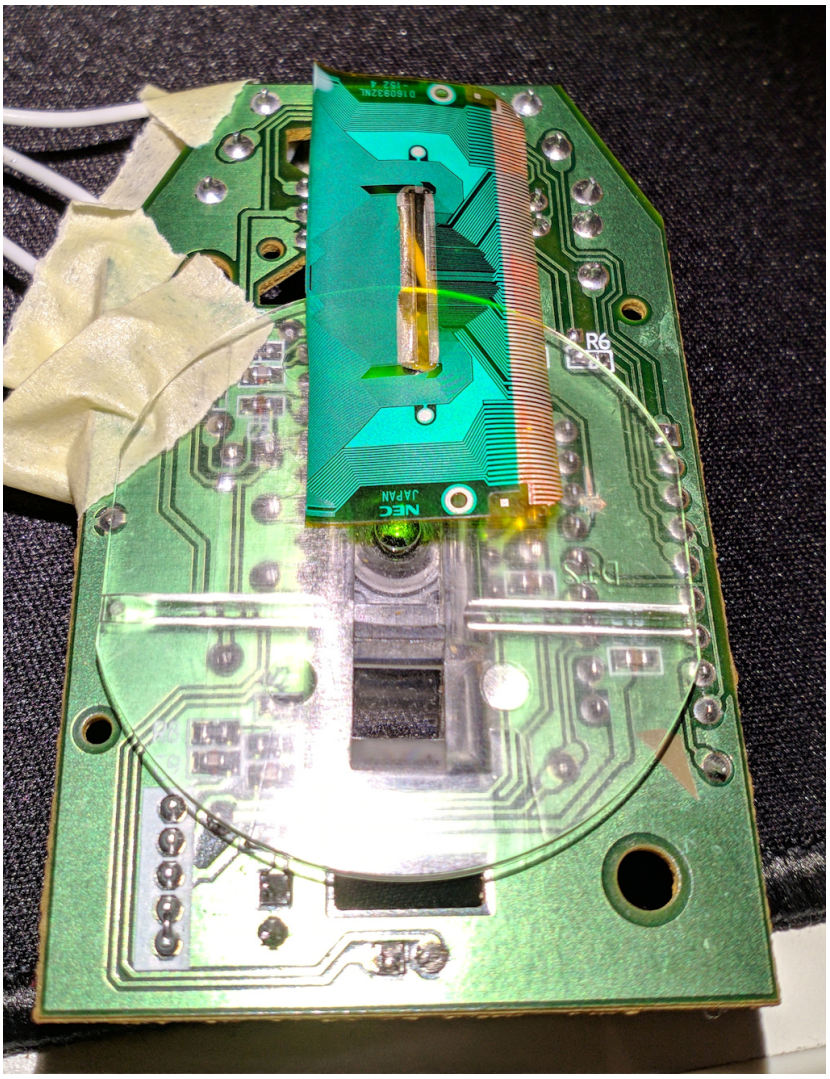
```
java -jar mousecam.jar
```

Käyttöliittymästä valitaan COM-portti, johon Arduino on kytketty ja painetaan “connect”, jolloin sarjaporttiyhteys muodostuu. Current sensor data -ikkuna näyttää hiiren tämänhetkisen näkymän. Hiirtä liikuttamalla voidaan nyt skannata sen alla olevaa pintaa, joka tulee näkyviin “scan area” -ikkunaan. Hiirtä liikuttaessa on huomioitavaa, että sensori ei havaitse kiertämistä, mikä vääristää skannausta. Ruudunpäivitysnopeus on melko hidas, joten hiiren liian nopea liikuttaminen vääristää myös kuvaa. “Frame dump width” -valinnalla (1-16) voi säätää kuinka monta saraketta sävykuvamatriisista luetaan. “Current sensor data” -laatikko päivittyy nopeammin pienemmillä arvoilla, koska mousecam-ohjelman täytyy noutaa sensorilta vähemmän dataa.[4]



Kuva 6. mousecam-ohjelman käyttöliittymä.

Kuvassa 7 nähdään skannausjärjestely. Sensorin alapuoli osoittaa ylöspäin ja linssi on asetettu sen päälle. Valaisunlähteenä normaalisti käytettävän punaisen ledin sijasta käytetään piirilevyn yläpuolelle sijoitettua ledivalaisinta. Kuvattava kohde on linssin päällä liikuteltava osittain läpinäkyvä ja kuvioitu kalvo, jossa on pienen resoluution kuvassa helposti tunnistettavissa olevia muotoja.



Kuva 7. Skannausjärjestely.

3.3. Tulosten selostus

Työssä onnistuttiin lukemaan sensorin sävykuva. Kuvissa 7 ja 8 nähdään sama "japan" -teksti ja raitakuvio. Kuvassa 6 nähdään paperille tulostetun numero yhden (1) alaosa. Kalvosta oli vaikea saada skannattua selkeää yhtenäistä aluetta, sillä kalvon pienikin kiertäminen tai liikuttaminen liian nopeasti aiheutti repeämiä ja vääristymiä kuvaan.

Kuva 8. Kuvankaappaus skannatusta alueesta.

4. POHDINTA

Tässä työssä toteutettiin ADNS-2051-sensorin käyttäminen kamerana, mutta sensorilla olisi mahdollista toteuttaa muitakin sovelluksia. Suunniteltu käyttötarkoitus on x- ja y-suunnan siirtymien seuraaminen. Sensori kykenee kuitenkin mittaamaan myös monia muita asioita, kuten nopeutta, valoisuutta ja kuvassa olevien piirteiden lukumäärää sekä ottamaan kuvia.

Kuten tuloksista näkee, ADNS-2051 -sensorin toiminnallisuus kamerana ei ole kummoinen matalan resoluution takia. Resoluutio ei ole kuitenkaan ongelma liikkeen seuraamisen kannalta, sillä HDNS-2100 -linssi tarkentaa kuvan hyvin pienelle alueelle, jolloin yksityiskohtia on mahdollista erottaa pienestä resoluutiosta huolimatta. Eräs vakio-optiikkaa käyttävä sovellus voisi olla vedenlaadun seuranta. SQUAL eli pinnanlaaturekisterin arvo kertoo kuvassa olevien piirteiden määrän, jolloin arvo korreloisi veden puhtauden kanssa.

Optiikan voisi vaihtaa kauemmaksi tarkentavaan, mikäli kuvan yksityiskohdilla ei ole väliä. SQUAL-rekisterin arvoa seuraamalla voisi toteuttaa liiketunnistimen, jolloin liike sensorin näkökentässä aiheuttaisi muutoksen kuvassa ja siten rekisterin arvossa. Tällainen liiketunnistin olisi melko alkeellinen ja altis monille häiriöille, mutta toimisi oletettavasti kohtuullisen luotettavasti hyvässä valaistuksessa.

Etsimällä aiheesta jo tehtyä tutkimusta voidaan löytää lukuisia muita sovelluksia, joista vain muutamia ovat esimerkiksi väärennettyjen kolikkojen havaitseminen[6], kaksiakselinen siirtymäsensori[7] ja pinnanmuotojen mittaus[8].

5. YHTEENVETO

Työn teoriaosuudessa käytiin läpi optisen tietokonehiiren toimintaperiaate ja tutustuttiin tarkemmin ADNS-2051-sensorin kanssa kommunikoimiseen ja sävykuvan lukemiseen. Optisessa hiiressä on kamera, joka ottaa peräkkäisiä kuvia sen alla olevasta pinnasta. Ledistä ja optiikasta koostuva valaisujärjestelmä valaisee hiiren alla olevan pinnan sekä tarkentaa kuvan sensorille. Digitaalinen signaaliprosessori laskee peräkkäin kaapattujen kuvien päällekkäisyydestä suhteellisia siirtymiä, jotka välitetään tietokoneelle USB:n välityksellä.

ADNS-2051-sensorin kanssa on mahdollista kommunikoida kaksijohtoisen sarjaporttikommunikaation avulla. Sarjaportin kautta voidaan lukea sensorin rekisterien sisältö ja osaan rekistereistä voidaan myös kirjoittaa, mikä mahdollistaa sensorin asetusten vaihtamisen. Sävykuvaa lukiessa kytketään päälle PixDump-toiminto kytkemällä Configuration_bits-rekisterin PixDump-bitti päälle, jolloin sensori dumppaa sävykuvan Data_Out_Upper- ja Data_Out_Lower-rekistereihin. Sävykuva luetaan rekistereistä ja tulostetaan sarjaportin kautta.

Työn käytännön osuudessa käytiin läpi hiirimodaukset, Arduinon ohjelmointi ja sävykuvan visualisointiin tarvittavien ohjelmien asentaminen tietokoneelle. Työ aloitettiin purkamalla hiiri ja poistamalla siitä työn kannalta turhat komponentit kuten rulla ja USB-johto. Seuraavaksi katkaistiin ADNS-2051-sensorin ja CP5928AM-mikrokontrollerin väliset SDIO- ja SCLK-johtimet piirilevyltä, sillä työssä tarvittiin vain itse sensoria. Nyt juotettiin sensorin tarvittaviin pinneihin johtimet, jotka yhdistettiin Arduinoon, johon oli lähetetty sävykuvaa lukeva mousecam-ohjelma. Tietokoneelle asennettiin 32-bittinen Java ja Javan RXTX-kirjasto sävykuvan visualisointiohjelman suorittamista varten. Sävykuva saatiin näkyviin.

Lopuksi pohditiin ADNS-2051-sensorille muita mahdollisia käyttötarkoituksia tietokonehiiren lisäksi sekä mainittiin muutama esimerkki sovelluksista, joista on jo tutkimusta. ADNS-2051-sensorin arvioitiin voivan toimia veden laadun mittauksessa ja liikkeen tarkkailussa SQUAL-rekisterin arvoa hyödyntämällä. Työssä käytetty ADNS-2051, tai mikä tahansa optisen hiiren sensori yleensäkin, on yllättävän monipuolinen sensori, jolle voi keksiä lukuisia sovelluksia.

LÄHTEET

[1] Teo Chiang Mei. "Understanding Optical Mice". *Avago Technologies* (2006) (ks. s. 6, 8).

[2] *Agilent ADNS-2051 Optical Mouse Sensor Data Sheet*. 5. maaliskuuta 2003. URL: https://www.alldatasheet.com/datasheet-pdf/pdf/102950/HP/ADNS-2051.html (viitattu 27. 03. 2020) (ks. s. 6–10).

[3] *PMW3360 Product Datasheet*. PixArt Imaging Inc., 6. huhtikuuta 2016. URL: https://www.pixart.com/products-detail/10/PMW3360DM-T2QU (viitattu 07. 04. 2020) (ks. s. 6, 8).

[4] *Optical mouse camera*. 12. huhtikuuta 2008. URL: https://www.bidouille.org/hack/mousecam (viitattu 02. 03. 2020) (ks. s. 10–12).

[5] *Download*. RXTX kirjaston latauslinkki. 28. kesäkuuta 2012. URL: http://rxtx.qbang.org/wiki/index.php/Download (viitattu 17. 03. 2020) (ks. s. 12).

[6] M. Tresanchez et al. "Using the optical mouse sensor as a two-Euro counterfeit coin detector". *Sensors* 9.9 (2009). cited By 22, s. 7083–7096. DOI: 10.3390/s90907083. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-70350029391&doi=10.3390%2fs90907083&partnerID=40&md5=71c5d448f7a95b0470033c55022c6522 (ks. s. 15).

[7] T.W. Ng. "The optical mouse as a two-dimensional displacement sensor". *Sensors and Actuators, A: Physical* 107.1 (2003). cited By 98, s. 21–25. DOI: 10.1016/S0924-4247(03)00256-5. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-0042281895&doi=10.1016%2fS0924-4247%2803%2900256-5&partnerID=40&md5=00b5b9d03a070c47f03fe105c7b57a8a (ks. s. 15).

[8] X. Wang ja K. Shida. "Surface shape analyzing device using optical mouse sensor". Teoksessa: cited By 2. 2009, s. 255–258. DOI: 10.1109/YCICT.2009.5382374. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-77950882016&doi=10.1109%2fYCICT.2009.5382374&partnerID=40&md5=3af5aefa210185df64d1e1e082faa4e6 (ks. s. 15).

LIITTEET

Liite 1: Arduino-ohjelma

Liite 1

```
#define SCLK 2
#define SDIO 3
#define PD 4

#define REG_PRODUCT_ID 0x00
#define REG_REVISION_ID 0x01
#define REG_MOTION 0x02
#define REG_DELTA_X 0x03
#define REG_DELTA_Y 0x04
#define REG_SQUAL 0x05
#define REG_AVERAGE_PIXEL 0x06
#define REG_MAXIMUM_PIXEL 0x07
#define REG_CONFIG_BITS 0x0A
#define REG_DATA_OUT_LOWER 0x0C
#define REG_DATA_OUT_UPPER 0x0D
#define REG_SHUTTER_LOWER 0x0E
#define REG_SHUTTER_UPPER 0x0F
#define REG_FRAME_PERIOD_LOWER 0x10
#define REG_FRAME_PERIOD_UPPER 0x11

int dumpWidth = 256; // Number of pixels to read for each frame.
byte frame[256];

void setup() {
  Serial.begin(115200);

  reset();
  byte productId = readRegister(REG_PRODUCT_ID);
  byte revisionId = readRegister(REG_REVISION_ID);
  Serial.print("Found productId ");
  Serial.print(productId, HEX);
  Serial.print(", rev. ");
  Serial.print(revisionId, HEX);
  Serial.println(productId == 0x02 ? " OK." : " Unknown productID. Carry
  ↪  on.");

  byte config = readRegister(REG_CONFIG_BITS);
  config |= B00000001; // Don't sleep (LED always powered on).
  writeRegister(REG_CONFIG_BITS, config);
}

void loop() {
  // Allows to set the dump window by sending the number of lines to read via
  ↪  the serial port.
```

```
43    if(Serial.available() > 0) {
44      dumpWidth = 16 * Serial.read();
45      dumpWidth = constrain(dumpWidth, 0, 256);
46    }
47
48    readRegister(REG_MOTION); // Freezes DX and DY until they are read or MOTION
      ↪  is read again.
49    char dx = readRegister(REG_DELTA_X);
50    char dy = readRegister(REG_DELTA_Y);
51    Serial.print("DELTA:");
52    Serial.print(dx, DEC);
53    Serial.print(" ");
54    Serial.println(dy, DEC);
55
56    if( dumpWidth > 0 )
57      dumpFrame();
58  }
59
60  void dumpFrame() {
61    byte config = readRegister(REG_CONFIG_BITS);
62    config |= B00001000; // PixDump
63    writeRegister(REG_CONFIG_BITS, config);
64
65    int count = 0;
66    do {
67      byte data = readRegister(REG_DATA_OUT_LOWER);
68      if( (data & 0x80) == 0 ) { // Data is valid
69        frame[count++] = data;
70      }
71    }
72    while (count != dumpWidth);
73
74    config = readRegister(REG_CONFIG_BITS);
75    config &= B11110111;
76    writeRegister(REG_CONFIG_BITS, config);
77
78    Serial.print("FRAME:");
79    for(int i = 0; i < dumpWidth; i++) {
80      byte pix = frame[i];
81      if( pix < 0x10 )
82        Serial.print("0");
83      Serial.print(pix, HEX);
84    }
85    Serial.println();
86  }
87
88  void reset() {
```

```
89    pinMode(SCLK, OUTPUT);
90    pinMode(SDIO, INPUT);
91    pinMode(PD, OUTPUT);
92    digitalWrite(SCLK, LOW);
93    digitalWrite(PD, HIGH);
94    delayMicroseconds(1);
95    digitalWrite(PD, LOW);
96  }
97
98  byte readRegister(byte address) {
99    pinMode (SDIO, OUTPUT);
100
101   for (byte i=128; i >0 ; i >>= 1) {
102     digitalWrite (SCLK, LOW);
103     digitalWrite (SDIO, (address & i) != 0 ? HIGH : LOW);
104     digitalWrite (SCLK, HIGH);
105   }
106
107   pinMode (SDIO, INPUT);
108
109   delayMicroseconds(100); // tHOLD = 100us min.
110
111   byte res = 0;
112   for (byte i=128; i >0 ; i >>= 1) {
113     digitalWrite (SCLK, LOW);
114     digitalWrite (SCLK, HIGH);
115     if( digitalRead (SDIO) == HIGH )
116       res |= i;
117   }
118
119   return res;
120 }
121
122 void writeRegister(byte address, byte data) {
123   address |= 0x80; // MSB indicates write mode.
124   pinMode (SDIO, OUTPUT);
125
126   for (byte i = 128; i > 0 ; i >>= 1) {
127     digitalWrite (SCLK, LOW);
128     digitalWrite (SDIO, (address & i) != 0 ? HIGH : LOW);
129     digitalWrite (SCLK, HIGH);
130   }
131
132   for (byte i = 128; i > 0 ; i >>= 1) {
133     digitalWrite (SCLK, LOW);
134     digitalWrite (SDIO, (data & i) != 0 ? HIGH : LOW);
135     digitalWrite (SCLK, HIGH);
```

```
136     }
137
138     delayMicroseconds(100); // tSWW, tSWR = 100us min.
139 }
```